IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR PATENT

BUFFER BYPASS CIRCUIT FOR REDUCING LATENCY IN INFORMATION TRANSFERS TO A BUS

Inventors: Hui Zhang,
Daniel Steinberg, and
Qi Bian.

FIELD OF THE INVENTION

[0001] The present invention generally relates to computer system buses and in particular, to a buffer bypass circuit for reducing latency in information transfers to a bus.

BACKGROUND OF THE INVENTION

[0002] Latency is time spent in an information transfer that is for activities other than the actual transferring of the information. Examples of latency in information transfers to a bus include time spent in requesting access to the bus, receiving and recognizing a grant to access the bus, and setting up the information transfer to the bus. The reduction of latency in information transfers to a bus is useful, because latency reduction generally provides improved system performance through more efficient bus utilization.

[0003] Pre-requests and bus parking are two techniques used to avoid latency associated with the process of requesting and being granted access to a bus. Using a pre-quest, access to the bus is requested and granted before,

rather than after, a current operation ends so that a next operation may occur without a request/grant delay. Using bus parking, a grant to access a bus is automatically given to a bus master (such as a central processing unit) whenever the bus is idle (i.e., the bus is not currently being used and there is no pending request to access the bus), so the bus master doesn't have to perform a grant/request procedure as long as the grant is active.

[0004] Buffers are also commonly used so that a device doesn't have to wait on a busy or otherwise unavailable bus before transferring data. Instead, the device transfers data to the buffer and consequently, is free to perform other tasks. When the bus is subsequently available, logic associated with the buffer manages the transfer of data from the buffer to the bus in a manner transparent to the device. The use of buffers in this fashion is especially useful when the device transferring information to the bus is a central processing unit.

[0005] Although each of these techniques, individually and/or in combination with one another, substantially reduce latency in information transfers to a bus, additional improvements are generally desirable whenever possible to further improve system performance through improved bus utilization.

OBJECTS AND SUMMARY OF THE INVENTION

[0006] The present invention is based upon the observation that latency in information transfers to a bus may be reduced under certain circumstances as described herein by bypassing a buffer normally used to buffer such

information transfers when the bus is not immediately available.

[0007] Accordingly, it is an object of the present invention to provide a method for bypassing a buffer for reducing latency in information transfers to a bus.

[0008] Another object is to provide a buffer bypass circuit for reducing latency in information transfers to a bus that complements other latency reducing circuitry when present.

[0009] These and additional objects are accomplished by the various aspects of the present invention, wherein briefly stated, one aspect is a method for reducing latency in information transfers to a bus, comprising: receiving an indication that information is to be transferred to a bus; reading a bus grant indication; writing the information to a buffer if the bus grant indication does not indicate that transfer of the information to the bus is allowed; and transferring the information to the bus if the buffer is empty and the bus grant indication indicates that transfer of the information to the bus is allowed.

[0010] Another aspect is a bus interface unit for reducing latency in information transfers from a device to a bus. The bus interface unit comprises a buffer having inputs coupled to the device; and logic configured to receive an indication from the device that information is to be transferred to the bus, read a bus grant indication, and cause the information to either be stored in the buffer if the bus grant information does not indicate that transfer of the information from the device to the bus is

allowed, or be transferred from the device to the bus if the buffer is empty and the bus grant indication indicates that transfer of the information to the bus is allowed.

[0011] Yet another aspect is a buffer bypass circuit for reducing latency in information transfers to a bus that is included in a computer system. The computer system includes the bus with access governed by a bus arbiter employing a bus parking scheme, and a buffer having inputs coupled to a device so that information to be transferred from the device to the bus is stored in the buffer if a grant indication generated by the bus arbiter indicates that the bus is unavailable for the transfer.

[0012] The buffer bypass circuit in this case comprises: a multiplexer having first inputs coupled to inputs to the buffer, second inputs coupled to outputs of the buffer, outputs coupled to the bus, and at least one select input for selectively coupling either the first or the second inputs to the outputs; and logic configured to provide control information to the at least one select input such that the first inputs are coupled to the outputs of the multiplexer if the buffer is empty and the grant indication indicates that the bus is available for transfer of the information to the bus.

[0013] Additional objects, features and advantages of the various aspects of the present invention will become apparent from the following description of its preferred embodiment, which description should be taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014] **FIG. 1** illustrates a block diagram of a computer system utilizing aspects of the present invention.

[0015] **FIG. 2** illustrates a block diagram of a prior art bus interface unit.

[0016] **FIGS. 3** and **4** respectively illustrate timing diagrams for write and read operations using the bus interface unit of **FIG. 2**.

[0017] **FIG. 5** illustrates a block diagram of a bus interface unit including a buffer bypass circuit utilizing aspects of the present invention.

[0018] **FIGS. 6** and **7** illustrates timing diagrams for write and read operations using the bus interface unit of **FIG. 5** utilizing aspects of the present invention.

[0019] **FIG. 8** illustrates a flow diagram of a method employed in the buffer bypass circuit for reducing latency in information transfers to a bus, utilizing aspects of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0020] **FIG. 1** illustrates, as an example, a block diagram of a Computer System **100**. A Bus **106** serves as main communication artery for various devices or components in the Computer System **100** to communicate with one another through information transfers. Among these devices are a Central Processing Unit (CPU) **101**, a Memory Controller **103** managing access to a Memory **102**, a Peripheral Controller

Interface (I/F) **108** managing and facilitating communications with a Peripheral **109**, an Ethernet I/F **111** managing and facilitating communications with other devices on a local area or other network, and a Direct Memory Access (DMA) Controller **114**.

[0021] Since only one device may use the Bus **106** at a time, a Bus Arbiter **107** is included in the Computer System **100** to manage access to the Bus **106** by arbitrating between competing requests issued by different devices. Although the CPU **101** must compete for access to the Bus **106** just like each of the other devices in the Computer System **100**, it is given some preferential treatment through a bus parking scheme employed by the Bus Arbiter **107**. Using this scheme, each time the Bus **106** becomes idle, the CPU **101** is granted access to the Bus **106** even though it may not have requested it at the time. This "bus parking" grant remains active until another device requests access to the Bus **106**, and that request is either granted or pending when the CPU **101** is not using the Bus **106**. Thus, the CPU **101** saves the time normally spent to request access to the Bus **106** whenever the bus **106** is idle.

[0022] The CPU **101** is coupled to the Bus **106** through a Bus Interface Unit (BIU) **105**. The BIU **105** serves as a bus bridge function between the Bus **106** and a CPU Bus **104** for any necessary protocol translation and/or timing synchronization activities, as well as performing latency reduction activities related to transferring information to and/or from the Bus **106** such as information buffering, pre-request/pre-grant processing, and buffer bypassing functions.

[0023] Each of the activities performed by the BIU **105** is performed in conventional BIU systems, except for the buffer bypassing function which is an aspect of the present invention. To appreciate the benefits of the buffer bypassing function, **FIGS. 2~4** are provided to illustrate the implementation and operation of a prior art BIU without the buffer bypassing function, and **FIGS. 5~8** illustrate the implementation and operation of the BIU **105** including a buffer bypassing circuit of the present invention.

[0024] Although a pre-request and pre-grant scheme is preferably performed in the BIU **105**, the pre-request/pre-grant activities are not discussed in reference to the following figures so as not to overly complicate the timing diagrams included therein. As should be readily appreciated, however, the buffer bypassing function of the present invention is fully compatible with, and its benefits complementary to, conventional pre-request/pre-grant schemes.

[0025] Referring now to **FIG. 2**, a Two-Entry Buffer **202** is provided to store two successive words of information that are to be transferred from the CPU **101** to the Bus **106**. The advantage of having the first entry to the Buffer **202** is so that the CPU **101** doesn't have to wait for the Bus **106** to become available if it is busy at the time that the CPU **101** initiates a transfer of information to the Bus **106**. With the assistance of Control Logic **201**, the CPU **101** can write information to be transferred to the Bus **106** into the Buffer **201** regardless of whether the Bus **106** is busy or not, and the Control Logic **202** manages subsequent transfer

of the written information from the Buffer **202** to the Bus **106** when the Bus **106** is available.

[0026] The advantage of having the second entry is so that the CPU **101** can perform at least a two word burst write to the Buffer **202** in cooperation with the Control Logic **201**. In particular, if the Bus **106** is unavailable at the time, then the CPU **101** can perform a two word burst write to the Buffer **202**. On the other hand, if the Bus **106** is available, then the CPU **101** can perform a burst write of more words of information to the Buffer **202** by continuing to write information into it while the written information is being concurrently transferred out to the Bus **106** by the Control Logic **201** with a one clock cycle offset between the writing in and transferring out of each word of information.

[0027] As can be readily appreciated, deeper (i.e., more than two entries) and wider (i.e., more than one word per entry) buffers may also be used in practicing the present invention with appropriate modifications to the operation and/or configuration of a buffer interface unit including the buffer.

[0028] The Control Logic **201** manages the writing of information into and out of the Buffer **202** through appropriate control signals (BFCN) to the Buffer **202**. Writing to the Buffer **202** is initiated after receiving an indication from the CPU **101** that information is to be transferred to the Bus **106**. Examples of such an indication are a write command (CPU_WR_COM) included in command information (CCO) if data is to be transferred to the Bus **106** (i.e., CDW), and a read command (CPU_RD_COM) included

in the command information (CCO) if data is to be received from the Bus **106** (i.e., BDR).

[0029] Transfer of information from the Buffer **202** to the Bus **106** is initiated by the Control Logic **201** after receiving a grant indication on a grant line (GNT/PARKING-GNT). Since the Arbiter **107** employs a bus parking scheme for the benefit of the CPU **101**, the Control Logic **201** checks to see if a parking grant (PARKING-GNT) is active on the grant line. This is done in the clock cycle that follows the first writing of information to the Buffer **202**. If the PARKING-GNT is active (i.e., the CPU **101** is granted access to the Bus **106**), then the Control Logic **201** starts transferring information from the Buffer **202** to the Bus **106** during the same clock cycle during which it checks the grant line.

[0030] An example of a prior art two word write operation from the CPU **101** to the Bus **106** is illustrated in **FIG. 3**. In a first clock cycle, the state (BUS_STATE) of the Bus **106** is idle. Therefore, the parking grant (PARKING-GNT) to the CPU **101** is active (defined in this case as being logic HIGH). In a second clock cycle, the CPU **101** initiates a transfer of information by providing an address (ADDR1) for a first word of the information on parallel address lines (CAD), data (DATA1) to be written to that address on parallel data lines (CDW), and appropriate commands for the first word such as a write command (WRITE) on parallel command lines (CCO). The Control Logic **201** then causes these three items comprising the first word to be written into the first entry of the Buffer **202** by

providing an appropriate command output on one or more control lines (BFCN) to the Buffer **202**.

[0031] In a third clock cycle, the CPU **101** provides the address (ADDR2) for a second word of the information to be transferred on parallel address lines (CAD), data (DATA2) to be written to that address on parallel data lines (CDW), and appropriate commands for the second word such as a write command (WRITE) on parallel command lines (CCO). The Control Logic **201** then causes these three items comprising the second word to be written into the second entry of the Buffer **202** by providing an appropriate control output on the one or more control lines (BFCN) to the Buffer **202**.

[0032] Concurrently in the third clock cycle, the Control Logic **201** also determines that the parking grant (PARKING-GNT) is still active, and causes the Buffer **202** to provide the address (ADDR1) for the first word of information to be transferred on parallel address lines (BAD), corresponding data (DATA1) for the first word on parallel data lines (BDW), and appropriate commands for the first word such as a write command (WRITE) on parallel command lines (BCO) of the Bus **106** from the first entry in the Buffer **202**. Since the Bus **106** is being used by the CPU **101** at this time, its state is changed from the idle state (IDLE) to a busy state (BUSY).

[0033] Note that although the Control Logic **201** could have determined in the second clock cycle whether the parking grant (PARKING-GNT) to the CPU **101** was still active, there is little advantage in doing so in that clock cycle, because the first word of the information would not

be available at that time for transfer out of the Buffer **202**.

[0034] In a fourth clock cycle, there are no more words of information to be transferred from the CPU **101**, but there is still the second word to transfer from the Buffer **202** to the Bus **106**. Therefore, the Control Logic **201** causes the Buffer **202** to provide the address (ADDR2) for the second word on parallel address (BAD), data (DATA2) for the second word on parallel data lines (BDW), and appropriate commands for the second word such as a write command (WRITE) on parallel command lines (BCO) of the Bus **106** from the second entry in the Buffer **202**.

[0035] Since the information transfer is now completed, the Control Logic **201** informs the Arbiter **107** that it is freeing the Bus **106** so that the bus state (BUS_STATE) goes back to its idle state (IDLE), and remains there until a seventh clock cycle when another device is granted access to and starts using the Bus **106**, thereby placing its bus state (BUS_STATE) in its busy state (BUSY). Although not shown, when this device is subsequently finished with the Bus **106**, the Arbitrator **107** will either grant the Bus **106** to a winning one of any pending bus requests at the time or automatically grant it to the CPU **101** if the Bus **106** is idle at the time.

[0036] An example of a prior art two word read operation is illustrated in **FIG. 4**. In a first clock cycle, the state (BUS_STATE) of the Bus **106** is idle, and the parking grant (PARKING-GNT) to the CPU **101** is active. In a second clock cycle, the CPU **101** initiates a transfer of information by providing an address (ADDR1) for a first

word of the information to be transferred on parallel address lines (CAD), and appropriate commands for the first word such as a read command (READ) on parallel command lines (CCO). The Control Logic **201** then causes these two items comprising the first word to be written into the first entry of the Buffer **202** by providing an appropriate control output on one or more control lines (BFCN) to the Buffer **202**.

[0037] In a third clock cycle, the CPU **101** provides the address (ADDR2) for a second word of the information to be transferred on parallel address lines (CAD), and appropriate commands for the second word such as a write command (WRITE) on parallel command lines (CCO). The Control Logic **201** then causes these two items comprising the second word to be written into the second entry of the Buffer **202** by providing an appropriate control output on the one or more control lines (BFCN) to the Buffer **202**.

[0038] Concurrently in the third clock cycle, the Control Logic **201** also determines that the parking grant (PARKING-GNT) to the CPU **101** is active, so it causes the Buffer **202** to provide the address (ADDR1) for the first word of information on parallel address (BAD), and appropriate commands for the first word such as a read command (READ) on parallel command lines (BCO) of the Bus **106** from the first entry in the Buffer **202**.

[0039] In a fourth clock cycle, there are no more words of information to be transferred from the CPU **101**, but there is still the second word of information to transfer from the Buffer **202** to the Bus **106**. Therefore, the Control Logic **201** causes the Buffer **202** to provide the address

(ADDR2) for the second word on parallel address lines (BAD), and appropriate commands for the second word such as a read command (READ) on parallel command lines (BCO) of the Bus **106** from the second entry in the Buffer **202**.

[0040] Also in the fourth clock cycle, data (DATA1) corresponding to the first word of information is read from the address (ADDR1) transferred to the Bus **106** in the previous (i.e., third) clock cycle, and provided on parallel data lines (BDR) of the Bus **106**. This information is not passed through a buffer, because the CPU **101** has been prior notified by the Control Logic **201** through control lines (CCN) that the data is coming so the CPU **101** is prepared to accept it. Likewise, in a fifth clock cycle, data (DATA2) corresponding to the second word of information is read from the address (ADDR2) transferred to the Bus **106** in the previous (i.e., fourth) clock cycle, and provided to the CPU **101** on the parallel data lines (BDR) of the Bus **106**.

[0041] Since the information transfer is now completed, the Control Logic **201** informs the Arbiter **107** that it is freeing the Bus **106** so that the bus state (BUS_STATE) goes back to its idle state (IDLE), and remains there until a seventh clock cycle when another device is granted access to and starts using the Bus **106**, thereby placing its bus state (BUS_STATE) into a busy state (BUSY).

[0042] **FIG. 5** illustrates, as an example, the addition of a Buffer Bypass Circuit to the system of **FIG. 2** that reduces latency in information transfers to the Bus **106**. In particular, by comparing the timing diagrams of **FIG. 3** corresponding to the system of **FIG. 2** to its counterpart

FIG. 6 corresponding to the system of **FIG. 5** with the added Buffer Bypass Circuit, a two word write operation from the CPU **101** to the Bus **106** is performed in one less clock cycle using the system of **FIG. 5**. Similarly, by comparing the timing diagrams of **FIG. 4** corresponding to the system of **FIG. 2** to its counterpart **FIG. 7** corresponding to the system of **FIG. 5** with the added Buffer Bypass Circuit, a two word read operation is also performed in one less clock cycle using the system of **FIG. 5**. Note that in **FIGS. 5~7**, the status of the grant line (GNT/PARKING-GNT) is checked in the same clock cycle that the Control Logic **501** receives an indication from the CPU **101** that information is to be transferred to the Bus **106**, because the information in this case is immediately available for transfer from the CPU Bus **104** to the Bus **106** at that time.

[0043] The Buffer Bypass Circuit comprises a Multiplexer (MUX) **503** and an enhanced Control Logic **501**. The MUX **503** allows selective bypassing of a Buffer **502** which otherwise operates and serves the same purpose as the Buffer **202** in **FIG. 2**. The enhanced Control Logic **501**, on the other hand, not only performs all of the functions described in reference to the Control Logic **201** in **FIG. 2**, but it also generates a control output provided to a select input (SEL) of the Multiplexer **503** in accordance with the flow diagram of a method illustrated in **FIG. 8**.

[0044] The Multiplexer **503** has a first set of inputs (designated by a "1") that are coupled to inputs of the Buffer **502**, and a second set of inputs (designated by a "0") that are coupled to outputs of the Buffer **502**. When the select input (SEL) to the MUX **503** is in a logic state

of "1", then the MUX **503** couples the first set of inputs to its outputs A, B and C so that an input word to the Buffer **501** is provided on the outputs. On the other hand, when the select input (SEL) to the MUX **503** is in a logic state of "0", then the MUX **503** couples the second set of inputs to its outputs A, B and C so that a word stored in the Buffer is provided on the outputs. In this second case, the circuit of **FIG. 5** operates identically as the circuit described in reference to **FIG. 2**. In the first case, the circuit of **FIG. 5** bypasses the Buffer **502** so that it is effectively eliminated from the operation of the circuit.

[0045] Referring to **FIG. 8**, in **801**, the Control Logic **501** receives an indication from the CPU **101** that information is to be transferred from the CPU **101** to the Bus **106**. During the same clock cycle, in **802**, the Control Logic **501** then reads a bus grant indication provided on the grant line (GNT/PARKING-GNT). In particular, if the Arbiter **107** employs a bus parking scheme, the Control Logic **501** checks to see at this point if a parking grant to the CPU **101** is active.

[0046] If the grant indication does not indicate that transfer of the information to the bus is allowed at that time (e.g., the parking grant to the CPU **101** is not active), then in **803**, the Control Logic **501** issues a request to access the Bus **106** on a request line (REQ) coupled directly to the Arbiter **107**, and in **804~803**, periodically continues to request such access until access is granted. The request may be issued in the same clock cycle at **801** or a subsequent clock cycle. Meanwhile during this time, the Control Logic **501** is also causing the

information to be stored in the Buffer **502** in the same manner as described in reference to **FIG. 2**. After receiving the grant (or before receiving the grant, but after determining that the parking grant to the CPU **101** was not active), the Control Logic **501** generates a control output which is provided to the select input (SEL) of the MUX **503** so that the select input (SEL) is in a logic state of "0".

[0047] On the other hand, if the grant indication checked in **802** indicates that transfer of the information to the bus is allowed (e.g., a parking grant to the CPU **101** is active), then **805~807** are performed in the same clock cycle as **801**. In **805**, the Control Logic **501** checks to see if the Buffer **502** is empty at the time. If the Buffer **502** is empty, then in **806**, the Control Logic **501** generates a control output which is provided to the select input (SEL) of the MUX **503** so that the select input (SEL) is in a logic state of "1". This effectively causes the Buffer **502** to be bypassed so that information transfers are performed directly from the CPU **101** to the Bus **106** and consequently, one clock cycle is saved by bypassing the Buffer **502**. On the other hand, if the Buffer **502** is not empty, then in **807**, the Control Logic **501** generates a control output which is provided to the select input (SEL) of the MUX **503** so that the select input (SEL) is in a logic state of "0". In this case, the system of **FIG. 5** operates in generally the same manner as the system of **FIG. 2**, with no saving of a clock cycle.

[0048] In the examples illustrated in **FIGS. 3~4** and **6~7**, it is assumed that the Bus **106** is in its idle state when

the CPU **101** initiates a transfer of information to the Bus **106**. In practice, this assumption may seldom occur if the Bus **106** is heavily utilized. Therefore, a frequently occurring case may be the Control Logic **201** finding the parking grant (PARKING-GNT) not active. In this case, the Control Logic **201** will issue a request (REQ) to the Arbiter **107** to access the Bus **106** while the first word of the information to be transferred is written into the first entry of the Buffer **202**. Consequently, in such a situation, there is little advantage in bypassing the buffer for subsequent word transfers to the Bus **106**.

[0049] However, if the Bus **106** is found to be in its idle state (IDLE) when the CPU **101** initiates a transfer of information to the Bus **106**, then latency in the transfer of information can be reduced by one clock cycle by bypassing the Buffer **202** and causing the CPU **101** to transfer the information directly to the Bus **106**. Even if the Bus **106** is found to be in its idle state only one out of every ten times when the CPU **101** initiates a transfer of information to the Bus **106**, that means bypassing the Buffer **202** each of those times results in saving one clock cycle every ten transfers of information from the CPU **101** to the Bus **106**. Such an increment to system performance is clearly worthwhile to implement in a high performance computer system.

[0050] Although the various aspects of the present invention have been described with respect to a preferred embodiment, it will be understood that the invention is entitled to full protection within the full scope of the appended claims.